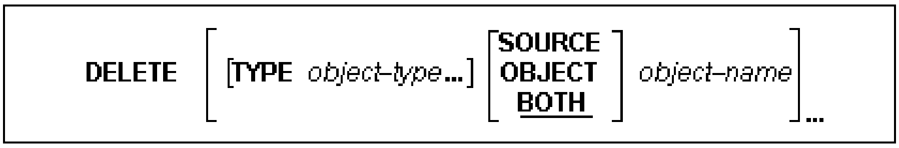
DELETE

Note:
This command is for Mainframes only. Aditionally, this command is not available via the command line in a remote development environment.



The DELETE command is used to delete Natural programming objects from the Natural system file.

You can only delete programming objects which are stored in your current library.

Note:
The source currently in the editor's work area is not affected by the DELETE command.

This section covers the following topics:

- object-name
- object-type
- Selection List
- Safeguard Against Accidental Deletion
- Examples

object-name

As *object-name*, you specify the name(s) of the object(s) to be deleted.

In addition, you can specify whether an object's source code, its object module, or both are to be deleted:

SOURCE	source code
OBJECT	object module
BOTH	both, source code and object module

If you make no specification, BOTH applies.

A SOURCE/OBJECT/BOTH specification applies for all subsequent object names that is, until the next SOURCE/OBJECT/BOTH specification.

To delete all objects whose names begin with a specific string of characters, you can use asterisk notation for the *object-name*.

object-type

In conjunction with asterisk notation for the *object-name*, you can also specify an *object-type* if you wish to delete only objects of a specific type.

The possible settings for *object-type* are the same as for the system command EDIT. In addition, you can specify the settings **X** (= global, local and parameter data areas) and **U** (= subprograms, subroutines and helproutines).

Note:
If you specify the full names of individual objects, you need not specify their object types.

Selection List

If you use asterisk notation, you will get a selection list, on which you then mark the object(s) to be deleted. For each object, you can determine whether to delete its source code, its object module, or both - by marking the object with the appropriate letter (**S**, **O** or **B**).

If you enter only the DELETE command itself, you will also get a selection list, containing all objects stored in your current library.

Safeguard Against Accidental Deletion

As a safeguard against accidental deletion, a window will automatically be displayed in which you have to confirm the deletion of an object by entering its name.

If you have specified or selected more than one object, an additional window will be displayed in which you can specify whether you wish to confirm the deletion for each object individually or whether all specified/selected objects are to be deleted without confirmation.

Examples

With this command, you delete three programming objects named TOM, DICK und HARRY:

```
DELETE TOM DICK HARRY
```

With this command, you delete the source and object module of the programming object JOHN, the sources of the programming objects PAUL and GEORGE, and the object module of the programming object RINGO:

```
DELETE JOHN SOURCE PAUL GEORGE OBJECT RINGO
```

With this command, you get a selection list of all programming objects in the current library:

```
DELETE
```

With this command, you get a selection list of the sources of all maps in the current library:

```
DELETE TYPE M SOURCE *
```

With this command, you get a selection list of all global, local and parameter data areas in the current library which are stored in source and/or object form and whose names begin with D:

```
DELETE TYPE GLA D*
```

With this command, you get a selection list of all programming objects in the current which are stored in object form and whose names begin with YYZ:

```
DELETE OBJECT YYZ*
```

With this command, you delete the source and object module of the maps TOM and DICK, the source of the map HARRY, the source of the program JOHN, and the object module of the program PAUL:

```
DELETE TYPE M TOM DICK SOURCE HARRY TYPE P JOHN OBJECT PAUL
```

If TOM, DICK and HARRY are not maps, or if JOHN and PAUL are not programs, they will not be deleted.